COMPUTABILITY (08/10/2024)

* Model of computation

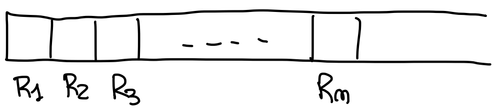
- → Turing machine
- → λ-calculus (Church)
- → partial recursive function (Gödel & Kleene)
- → canonical deduction systems (Post)
- → URM (Unlimited Register Machine)
- ⋮

Church-Turing thesis

A function is computable by an effective procedure   if and only if   it is computable by a Turing machine

Unlimited Register Machine

→ memory (unbounded)



$r_n \in \mathbb{N}$

→ executes a program : finite list of instructions

I_1
I_2
⋮
I_s

taken from a finite instruction set

Arithmetic instructions

→ zero	$Z(n)$	$r_n \leftarrow 0$
→ successor	$S(n)$	$r_n \leftarrow r_n + 1$
→ transfer	$T(m, n)$	$r_n \leftarrow r_m$

Jump

$J(m, n, t)$ $\quad r_m = r_n$? $\quad$ yes $\Rightarrow$ Jump to I_t

no $\Rightarrow$ continue with next instr.

* ## Computation

initially : $\begin{cases} \text{start configuration of registers} \\ \text{executes } I_1 \end{cases}$

continue according to the semantics

terminate : if we execute a non-existing instruction

→ last instruction , move to next which does not exist

→ jump out of the program

Example

		R_1	R_2	R_3	
I_1	$J(2,3,5)$	1	2	0	
I_2	$S(1)$	2	2	0	
I_3	$S(3)$	2	2	1	
I_4	$J(1,1,1)$	3	2	2	

terminates

* A computation might not terminate

I_1 $\quad J(1,1,1)$

* **Notation** : Given $a_1, a_2, \ldots \in \mathbb{N}$ and URM-program P

$P(a_1, a_2, \ldots)$ denotes the computation of P from $a_1, a_2, \ldots$

$\begin{cases} P(a_1, a_2, \ldots)\downarrow & \text{computation eventually terminates} \\ P(a_1, a_2, \ldots)\uparrow & \text{" diverges} \end{cases}$

Given $a_1, \dots, a_k \in \mathbb{N}$

$P(a_1, a_2, \dots, a_k)$ represents $P(a_1, \dots, a_k, 0, 0, \dots)$

$P(a_1, \dots, a_k) \downarrow a$ " $P(a_1, \dots, a_k)\downarrow$
and in final configuration $r_1 = a$

URM-computable function

Given $f : \mathbb{N}^k \to \mathbb{N}$ (possibly partial) is URM-computable if there is P URM-program such that $\forall (a_1, \dots, a_k) \in \mathbb{N}$ $\forall a \in \mathbb{N}$

$P(a_1, \dots, a_k) \downarrow a$ iff $(a_1, \dots, a_k) \in dom(f)$

and $f(a_1, \dots, a_k) = a$

$$\mathcal{C}^{(k)} = \{ f \mid f : \mathbb{N}^k \to \mathbb{N} \text{ URM-computable} \}$$

$$\mathcal{C} = \bigcup_{k \geq 1} \mathcal{C}^{(k)}$$

Example

* $f : \mathbb{N}^2 \to \mathbb{N}$

$f(x, y) = x + y$

R_1	R_2	R_3	
x	y	0	...
$x+k$		k	

$\rightsquigarrow$

$x+y$	---

```
LOOP: J(2, 3, END)      // y = k ?
      S(1)              // x = x+1
      S(3)              // k = k+1
      J(1, 1, LOOP)
END:
```

* $g : \mathbb{N} \to \mathbb{N}$

$$g(x) = x \dot{-} 1 = \begin{cases} 0 & \text{if } x = 0 \\ x - 1 & \text{otherwise} \end{cases}$$

R_1	R_2	R_3
x	0	0
	$k+1$	k

```
        J(1,2, END)          // x = 0 ?
        S(2)
LOOP :  J(1, 2, RES)         // x = K+1
        S(2)
        S(3)
        J(1,1,LOOP)
RES  :  T(3,1)
END  :
```

* $h : \mathbb{N} \to \mathbb{N}$

$$h(x) = \begin{cases} x/2 & \text{if } x \text{ even} \\ \uparrow & \text{otherwise} \end{cases}$$

R_1	R_2	R_3
x	0	0
	K	$2K$

```
LOOP :  J(1,3, RES)
        S(2)
        S(3)
        S(3)
        J(1,1, LOOP)
RES  :  T(2,1)
```

* <u>Function computed by a program</u>

given a program P and $K \geq 1$

$$f_P^{(K)} : \mathbb{N}^K \to \mathbb{N}$$

$$f_P^{(K)}(a_1, , a_K) = \begin{cases} a & \text{if } P(a_1, , a_K) \downarrow a \\ \uparrow & \text{if } P(a_1, , a_K) \uparrow \end{cases}$$

<u>Question</u> : given $f : \mathbb{N}^K \to \mathbb{N}$ how many programs computing f ?

infinitely many or none

EXERCISE : Consider URM$^-$ machine without $T(m,n)$

$\mathcal{C}^-$ = class of functions computed by URM$^-$

$$\mathcal{C}^- \stackrel{?}{=} \mathcal{C}$$

$\subseteq$

$\supseteq$ $\rightsquigarrow$ $T(m,n)$ $\rightsquigarrow$

$$\boxed{\begin{array}{ll} & Z(n) \\ \text{LOOP}: & J(m,n,\text{DONE}) \\ & S(n) \\ & J(1,1,\text{LOOP}) \end{array}}$$

proof

($\mathcal{C}^- \subseteq \mathcal{C}$) let $f: \mathbb{N}^k \to \mathbb{N}$ s.t. $f \in \mathcal{C}^-$, i.e. there is P URM$^-$-program s.t. $f_P^{(k)} = f$

Just observe that P is also a URM-program, hence $f = f_P^{(k)} \in \mathcal{C}$

($\mathcal{C} \subseteq \mathcal{C}^-$) let $f \in \mathcal{C}$ $f: \mathbb{N}^k \to \mathbb{N}$ and let P be a URM-program for f $f_P^{(k)} = f$

Assume : P is well-formed, if it terminates it does at the instruction following the last one

without loss of generality ⊛

We show that there is P' URM$^-$ program such that

$$f_{P'}^{(k)} = f_P^{(k)} = f$$

by induction on h = number of instructions $T(m,n)$ in P

($h=0$) P has no transfer instructions hence it is already URM$^-$-program and thus $P' = P$ works.

($h \to h+1$) let P a URM program with $h+1$ transfer instructions

$$P \left\{ \begin{array}{ll} I_1 & \\ \vdots & \\ I_t & T(m,n) \\ \vdots & \\ I_s & \end{array} \right.$$

we transform it in P''

$$P'' \begin{cases} I_1 \\ \vdots \\ I_t \quad J(1,1,SUB) \\ \vdots \\ I_s \\ I_{s+1} : \; J(1,1,END) \\ SUB : \; Z(m) \\ LOOP : \; J(m,n,t+1) \\ \qquad\quad S(m) \\ \qquad\quad J(1,1,LOOP) \\ END : \end{cases}$$

Now

- $f_{P''}^{(k)} = f_P^{(k)}$ (*)
- P'' has one T-instruction and so h transfer instructions

  hence by inductive hyp. there is P' URM$^-$-program such that $f_{P'}^{(k)} = f_{P''}^{(k)}$ (*)

By putting things together

$$f_P^{(k)} = f_{P''}^{(k)} = f_{P'}^{(k)}$$

where P' is a URM$^-$-program

$\square$

(*)
Note : for every URM program P there is a well-formed program P' s.t. $f_P^{(k)} = f_{P'}^{(k)} \quad \forall k$

idea

$$P \begin{cases} I_1 \\ \vdots \\ \qquad J(m,n,t) \\ \\ I_s \end{cases} \quad \rightsquigarrow \quad \text{if } t>s \text{, replace it with } J(m,n,s+1)$$

Exercise : variant of URM machine

URM^s $\qquad \cancel{T(m,n)} \qquad T^s(m,n)$ swaps $r_m \leftrightarrow r_n$

$$\mathcal{C}^s \stackrel{?}{=} \mathcal{C}$$

Exercise : Consider $URM^=$ without jump

$$\mathcal{C}^= \stackrel{?}{=} \mathcal{C}$$

$\subsetneq$?

$\uparrow$ characterise functions in $\mathcal{C}^=$